Inactivation Decoding of LT and Raptor Codes: Analysis and Code Design

Francisco Lázaro, *Student Member, IEEE*, Gianluigi Liva, *Senior Member, IEEE*, Gerhard Bauch, *Fellow, IEEE*

***Abstract*—In this paper we analyze LT and Raptor codes under inactivation decoding. A first order analysis is introduced, which provides the expected number of inactivations for an LT code, as a function of the output distribution, the number of input symbols and the decoding overhead. The analysis is then extended to the calculation of the distribution of the number of inactivations. In both cases, random inactivation is assumed. The developed analytical tools are then exploited to design LT and Raptor codes, enabling a tight control on the decoding complexity vs. failure probability trade-off. The accuracy of the approach is confirmed by numerical simulations.**

***Index Terms*—Fountain codes, LT codes, Raptor codes, erasure correction, maximum likelihood decoding, inactivation decoding.**

I. Introduction

FOUNTAIN codes [3], [4] provide an efficient solution for data delivery to large user populations over broadcast channels. The result is attained by encoding the data object (e.g., a file) through an (n, k) linear code, where the number of *output* (i.e., encoded) symbols n can grow indefinitely, enabling a simple rate adaptation to the channel conditions. Due to this, fountain codes are often regarded as *rateless* codes. Fountain codes were originally conceived for transmission over erasure channels. In practice, different users experience a different channel quality resulting in a different erasure probability. When an efficient fountain code is employed, after a user has received $m = k+\delta$ output symbols, with δ small, the original input symbols can be recovered with high probability.

Luby transform (LT) codes were introduced in [5] as a first example of practical fountain codes. In [5] an iterative erasure decoding algorithm was introduced that performs remarkably well when the number of input symbols k is large. Raptor codes [6]–[8] address some of the shortcomings of LT codes. A Raptor code is a serial concatenation of an (h, k) outer linear block code with an inner LT code.

Most of the literature on LT and Raptor codes considers iterative decoding (see e.g. [9]–[15]). In [9], [14], [16] LT codes under iterative decoding were analyzed using a dynamic programming approach. This analysis models the iterative decoder as a finite state machine and it can be used to derive the probability of decoding failure of LT codes under iterative decoding. Iterative decoding is particularly effective for large input block sizes, with k at least in the order of several tens of thousands symbols [8]. However, in practice, moderate and small values of k are often used, due to memory limitations at the receiver side, or due to the fact that the piece of data to be transmitted is small. For example, the recommended value of k for the Raptor codes standardized in [17] is between 1024 and 8192 symbols. In this regime, iterative decoding of LT and Raptor codes turns to be largely sub-optimum. In actual implementations, different decoding algorithms are used. In particular, an efficient maximum likelihood (ML) decoding algorithm usually referred to as *inactivation decoding* [18], [19] is frequently employed.

Inactivation decoding belongs to a large class of Gaussian elimination algorithms tailored to the solution of large sparse linear systems [20]–[25]. The algorithm can be seen as an extension of iterative decoding, where whenever the iterative decoding process stops, a variable (input symbol) is declared as *inactive* (i.e., removed from the equation system) and iterative decoding is resumed. At the end of the process, the inactive variables have to be solved using Gaussian elimination. If a unique solution is found, it is possible to recover all the remaining input symbols by back-substitution (i.e., using iterative decoding). A few works addressed the performance of LT and Raptor codes under inactivation decoding. The decoding failure probability of several types of LT codes was thoroughly analyzed via tight lower and upper bounds in [26]–[29]. In [30] the distance spectrum of *fixed-rate* Raptor codes is characterized, which enables the evaluation of their performance under ML erasure decoding (e.g., via upper union bounds [31], [32]). Upper bounds on the failure probability for binary and non-binary Raptor codes are introduced in [33]. However, none of these works addressed inactivation decoding from a complexity viewpoint. In fact, the complexity of inactivation decoding grows with the third power of the number of inactivations [24]. It is hence of large practical interest to develop a code design which leads (on average) to

Francisco Lázaro and Gianluigi Liva are with the Institute of Communications and Navigation of the German Aerospace Center (DLR), Muenchner Strasse 20, 82234 Wessling, Germany. Email:{Francisco.LazaroBlasco, Gianluigi.Liva}@dlr.de.

Gerhard Bauch is with the Institute for Telecommunication, Hamburg University of Technology, Hamburg, Germany. E-mail: Bauch@tuhh.de.

Corresponding Address: Francisco Lázaro, KN-SAN, DLR, Muenchner Strasse 20, 82234 Wessling, Germany. Tel: +49-8153 28-3211, Fax: +49-8153 28-2844, E-mail: Francisco.LazaroBlasco@dlr.de.

This work has been presented in part at the 54th Annual Allerton Conference on Communication, Control, and Computing, Monticello, Illinois, USA, September 2015 [1], and at the 2014 IEEE Information Theory Workshop, Hobart, Tasmania, November 2014 [2].

This work has been accepted for publication in IEEE Transactions on Communications, DOI: 10.1109/TCOMM.2017.2715805

a small number of inactivations.

In [25], inactivation decoding of low-density parity-check (LDPC) codes was analyzed, providing a reliable estimate of the expected number of inactivations. The approach of [25] relies on bridging the inactivation decoder with the Maxwell decoder of [34]. By establishing an equivalence between inactive and guessed variables, it was shown how to exploit the Area Theorem [35] to derive the average number of inactivations. Though, the approach of [25] cannot be extended to LT and Raptor codes due to their rateless nature. In [36] the authors proposed a new degree distribution design criterion to design the LT component of Raptor codes assuming inactivation decoding. However, when this criterion is employed there is not a direct control on the average output degree. In [37], the authors present a finite length analysis of batched sparse codes under inactivation decoding that provides the expected number of inactivations needed for decoding.

In this paper we provide a thorough analysis of LT and Raptor codes under inactivation decoding. A first order analysis is introduced, which allows estimating the expected number of inactivations for an LT code, as a function of the output distribution, the number of input symbols and the decoding overhead. The analysis is then extended to the calculation of the distribution of the number of inactivations. In both cases, random inactivation is assumed. The developed analytical tools are then exploited to design LT and Raptor codes, enabling a tight control on the decoding complexity vs. failure probability trade-off. The work presented in this paper is an extension of the work in [1]. In particular, the different proofs in [1] have been modified and extended. Furthermore, in this work we not only consider LT codes but also a class of Raptor Codes. An analysis based on a Poisson assumption approximation is presented in the Appendix[1].

The rest of the paper is structured as follows. In Section II we present the system model considered in this paper. Section III contains a detailed description of inactivation decoding applied to LT codes. In Section IV we focus on the analysis of LT codes under inactivation decoding. Section V shows how to exploit the analysis of the LT component of a Raptor code to estimate the number of inactivations for the Raptor code. A Raptor code design methodology is presented too. The conclusions are presented in Section VI.

II. Preliminaries

Let $\boldsymbol{v} = (v_1, v_2, ..., v_k)$ be the k input symbols, out of which the LT encoder generates the output symbols $\boldsymbol{c} = (c_1, c_2, ..., c_n)$, where n can grown indefinitely. Each output symbol is generated by first sampling a degree d from an output degree distribution $\Omega = (\Omega_1, \Omega_2, \Omega_3, \dots \Omega_{d_{\max}})$, where $d_{\max}$ is the maximum output degree. In the following, we will make use of a polynomial representation of the degree distribution in the form

$$\Omega(\mathtt{x}) := \sum_d \Omega_d \mathtt{x}^d$$

where $\mathtt{x}$ is a dummy variable. Then, d distinct input symbols are selected uniformly at random and their x-or is computed to generate the output symbol. The relation between output and input symbols can be expressed as

$$\boldsymbol{c} = \boldsymbol{v}\boldsymbol{G}$$

where $\boldsymbol{G}$ is the generator matrix of the LT code.

The output symbols are transmitted over a binary erasure channel (BEC) at the output of which the receiver collects $m = k + \delta$ output symbols, where δ is referred to as absolute receiver overhead. Similarly, the relative receiver overhead is defined as $\epsilon := \delta/k$. Denoting by $\boldsymbol{y} = (y_1, y_2, \dots, y_m)$ the m received output symbols and by $\mathcal{I} = \{i_1, i_2, \dots, i_m\}$ the set of indices corresponding to the m non-erased symbols, we have

$$y_j = c_{i_j}.$$

This allows us to express the dependence of the received output symbols on the input symbols as

$$\boldsymbol{G}_0^\mathsf{T} \boldsymbol{v}^\mathsf{T} = \boldsymbol{y}^\mathsf{T} \tag{1}$$

with $\boldsymbol{G}_0$ given by the m columns of $\boldsymbol{G}$ with indices in $\mathcal{I}$. ML erasure decoding requires the solution of (1). Note that decoding is successful (i.e., the unique solution can be found) if and only if $\mathrm{rank}\,[\boldsymbol{G}_0] = k$.

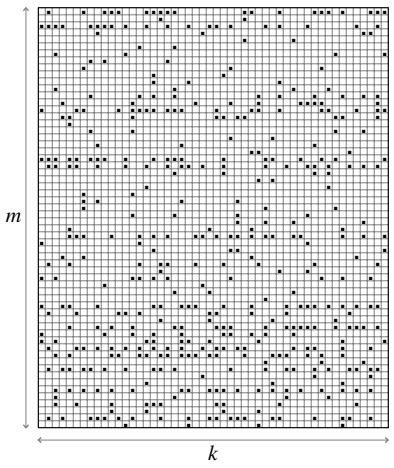
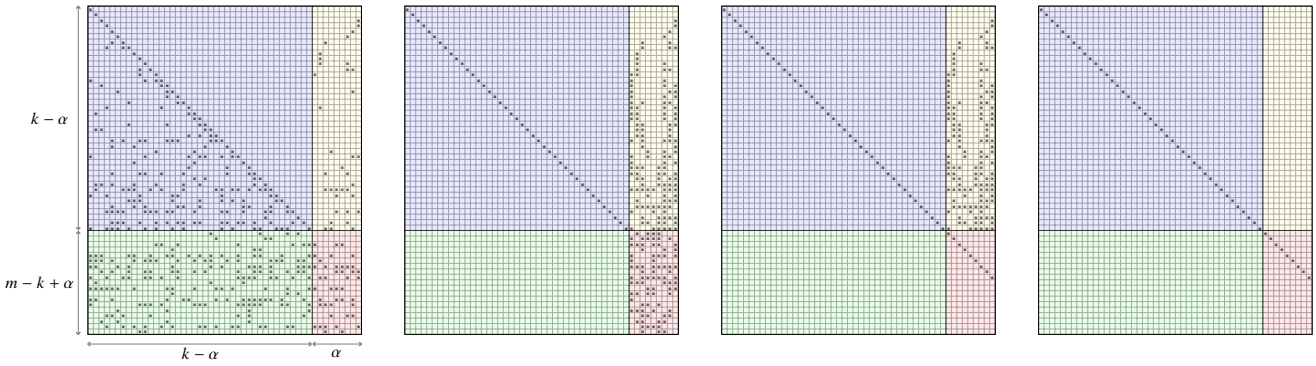


Fig. 1. Structure of the matrix $\boldsymbol{G}_0^\mathsf{T}$.

III. Inactivation Decoding of LT Codes

Inactivation decoding is an efficient ML erasure decoding algorithm, which can be used to solve the system of equations in (1). We will illustrate inactivation decoding by means of an example and with the aid of Figures 1 and 2. In the example, we fix $k = 50$ and $m = 60$. The structure of $\boldsymbol{G}_0^\mathsf{T}$ is given in Figure 1 (in the figure, dark squares represent the non-zero elements of $\boldsymbol{G}_0^\mathsf{T}$). Inactivation decoding consists of 4 steps.

[1]In the Appendix we provide a simplified analysis, which, though related to the one in [2], is fully developed here on the basis of the analysis provided in Section IV of this paper. The simplified analysis is based on a Poisson approximation, which is analyzed in depth in the Appendix, discussing its limitations and comparing the analytical results with Monte Carlo simulations.



(a) Triangulation process. (b) Zero matrix procedure. (c) Gaussian elimination. (d) Back-substitution.

Fig. 2. Structure of $\boldsymbol{G}_0^{\mathsf{T}}$ as inactivation decoding proceeds.

Step 1 (Triangulation). Initially, $\boldsymbol{G}_0^{\mathsf{T}}$ is put in an approximate lower triangular form by means of column and row permutations only. Given that no operation is performed on the rows or columns of $\boldsymbol{G}_0^{\mathsf{T}}$, the density of $\boldsymbol{G}_0^{\mathsf{T}}$ is preserved. At the end of triangulation process, $\boldsymbol{G}_0^{\mathsf{T}}$ can be partitioned in four submatrices as

$$\left[\begin{array}{c|c} \boldsymbol{A} & \boldsymbol{D} \\ \hline \boldsymbol{B} & \boldsymbol{C} \end{array}\right]$$

and as depicted in Figure 2a. In the upper left part we have the $(k-\alpha)\times(k-\alpha)$ lower triangular matrix $\boldsymbol{A}$. The matrix under submatrix $\boldsymbol{A}$ is denoted by $\boldsymbol{B}$ and it has dimension $(m-k+\alpha)\times(k-\alpha)$. The upper right part is given by the $(k-\alpha)\times\alpha$ submatrix $\boldsymbol{D}$, while the lower right submatrix is denoted by $\boldsymbol{C}$ and it has dimension $(m-k+\alpha)\times\alpha$. The α rightmost columns of $\boldsymbol{G}_0^{\mathsf{T}}$ (corresponding to matrices $\boldsymbol{C}$ and $\boldsymbol{D}$) are usually referred to as inactive columns. Note that the manipulation of $\boldsymbol{G}_0^{\mathsf{T}}$ is typically performed through *inactivation* or *pivoting* algorithms (see e.g. [24], [38]), which aim at reducing the number of inactive columns. In the rest of this work, we will assume that the use of the random inactivation algorithm [38], as it will be detailed later in the section.

Step 2 (Zero matrix procedure). Matrix $\boldsymbol{A}$ is put in a diagonal form and matrix $\boldsymbol{B}$ is zeroed out by means of row sums. As a result, on average the density of the matrices $\boldsymbol{C}$ and $\boldsymbol{D}$ increases (Figure 2b).

Step 3 (Gaussian elimination). Gaussian elimination (GE) is applied to solve the system of equations associated with $\boldsymbol{C}$. Being $\boldsymbol{C}$ in general dense, the complexity of this step is cubic in the number of inactive columns α. As observed in [24], this step drives, for large equation systems, the complexity of inactivation decoding. After performing GE, matrix $\boldsymbol{G}_0^{\mathsf{T}}$ has the structure shown in Figure 2c.

Step 4 (Back substitution). If Step 3 succeeds, after determining the value of the inactive variables (i.e., of the input symbols associated with the inactive columns), back-substitution is applied to compute the values of the remaining variables in $\boldsymbol{v}$. This is equivalent to setting to zero all elements of matrix $\boldsymbol{D}$ in Figure 2c. At the end $\boldsymbol{G}_0^{\mathsf{T}}$ shows a diagonal structure as shown in Figure 2d, and all input symbols are recovered.

Recalling that a unique solution to the system of equations exists if and only if $\boldsymbol{G}_0$ is full rank, we have that decoding succeeds if and only if the rank of $\boldsymbol{C}$ at Step 3 equals the number of inactive variables.

Given that the number of inactive variables is determined by the triangulation step, and that the GE on $\boldsymbol{C}$ dominates the decoding complexity for large k, a first analysis of inactivation decoding can be addressed by focusing on the triangulation procedure.

A. Bipartite Graph Representation of Inactivation Decoding

We introduce next a bipartite graph representation of the LT code from the receiver perspective. The graph comprises $m+k$ nodes, divided in two sets. The first set consists of k *input symbol nodes*, one per input symbol, while the second set consists of m *output symbol nodes*, one per received output symbol. We denote the set of input symbol nodes as $\mathscr{V}$, and the input symbol nodes in $\mathscr{V}$ as $v_1, v_2, \ldots, v_k$. Similarly, we denote the set of output symbol nodes as $\mathscr{Y}$ with output symbol nodes denoted by $y_1, y_2, \ldots, y_m$. Here, we purposely referred to input and output symbol nodes with the same notation used for their respective input and output symbols to emphasize the correspondence between the two sets of nodes and the sets of input and received output symbols. For simplicity, in the following we will refer to input (output) symbol nodes and to input (output) symbols interchangeably. An input symbol node v_i is connected by an edge to an output symbol node y_j if and only if the corresponding input symbol contributes to the generation of the corresponding output symbol, i.e., if and only if the (i,j) element of $\boldsymbol{G}_0$ is equal to 1. We denote by $\deg(v)$ (or $\deg(y)$) the degree of a node v (or y), i.e., the number of its adjacent edges.

Triangulation can be modelled as an iterative pruning of the bipartite graph of the LT code. At each iteration, a reduced graph is obtained, which corresponds to a sub-graph of the original LT code graph. The sub-graph involves only a subset of the original input symbols, and their neighboring output symbols. The input symbols in the reduced graph will be referred to as *active* input symbols. We shall use the term *reduced* degree of a node to refer to the degree of a node in the reduced graph. Obviously, the reduced degree of a node is less than or equal to its original degree. We denote by $\mathrm{red}(v)$

(or red(y)) the reduced degree of a node v (or y). Let us now introduce some additional definitions that will be used to model the triangulation step.

Definition 1 (Ripple)**.** *We define the ripple as the set of output symbols of reduced degree 1 and we denote it by $\mathscr{R}$.*

The cardinality of the ripple is denoted by $\mathtt{r}$ and the corresponding random variable as $\mathtt{R}$.

Definition 2 (Cloud)**.** *We define the cloud as the set of output symbols of reduced degree $d \geq 2$ and we denote it by $\mathscr{C}$.*

The cardinality of the cloud is denoted by $\mathtt{c}$ and the corresponding random variable as $\mathtt{C}$.

Initially, all input symbols are marked as active, i.e., the reduced sub-graph coincides with the original graph. At every step of the process, one active input symbol is marked as either *resolvable* or *inactive* and leaves the graph. After k steps no active symbols are present and triangulation ends. In order to keep track of the steps of the triangulation procedure, the temporal dimension will be added through the subscript u. This subscript u corresponds to the number of active input symbols in the graph. Given the fact that the number of active input symbols decreases by 1 at each step, triangulation will start with $u = k$ active input symbols and it will end after k steps with $u = 0$. Therefore, the subscript decreases as the triangulation procedure progresses. Triangulation with random inactivations works as follows. Consider the transition from u to $u - 1$ active input symbols. Then,

- If the ripple $\mathscr{R}_u$ is not empty ($\mathtt{r}_u > 0$), the decoder selects an output symbol $y \in \mathscr{R}_u$ uniformly at random. The only neighbor of y is marked as resolvable and leaves the reduced graph, while all edges attached to it are removed.
- If the ripple $\mathscr{R}_u$ is empty ($\mathtt{r}_u = 0$), one of the active input symbols v is chosen uniformly at random[2] and it is marked as inactive, leaving the reduced graph. All edges attached to v are removed.

The input symbols marked as inactive are recovered using Gaussian elimination (step 3). After recovering the inactive input symbols, the remaining input symbols, those marked as resolvable, can be recovered by iterative decoding (back substitution in step 4).

Example 1. *We provide an example for an LT code with $k = 4$ input symbols and $m = 4$ output symbols (Figure 3).*

i. *Transition from $u = 4$ to $u = 3$. Initially, there are two output symbols in the ripple ($\mathtt{r}_4 = 2$) (Figure 3a). Thus, one of the input symbols in the ripple is randomly selected and marked as resolvable. In this case symbol v_1 is selected and all its adjacent edges are removed. The graph obtained after the transition is shown in Figure 3b. Observe that the nodes y_1 and y_4 have left the graph since their reduced degree is now zero.*
ii. *Transition from $u = 3$ to $u = 2$. As shown in Figure 3b the ripple is now empty ($\mathtt{r}_3 = 0$). Thus, an inactivation has to take place. Node v_2 is chosen (according to a random selection) and is marked as inactive. All edges attached to v_2 are removed from the graph. As a consequence, the nodes y_2 and y_3 that were in the cloud $\mathscr{C}_3$ enter the ripple $\mathscr{R}_2$ (i.e., their reduced degree is now 1), as shown in Figure 3c.*
iii. *Transition from $u = 2$ to $u = 1$. Now the ripple is not empty ($\mathtt{r}_2 = 2$). The input symbol v_3 is selected (again, according to a random choice) from the ripple and is marked as resolvable. All its adjacent edges are removed. The nodes y_2 and y_3 leave the graph because their reduced degree becomes zero (see Figure 3d).*
iv. *Transition from $u = 1$ to $u = 0$. The ripple is now empty (Figure 3d). Hence, an inactivation takes place: node v_4 is marked as inactive and the triangulation procedure ends.*

Note that in this example input symbol v_4 has no neighbors, i.e., the matrix $\boldsymbol{G}_0$ is not full rank and decoding fails.

IV. Analysis under Random Inactivation

In this section we analyze the triangulation procedure with the objective of determining the average number of inactivations, i.e., the expected number of inactive input symbols at the end of the triangulation process. We will then extend the analysis to obtain the probability distribution of the number of inactivations.

A. Average Number of Inactivations

Following [9], [14], [16], we model the decoder as a finite state machine with state at time u given by the cloud and the ripple sizes at time u, i.e.

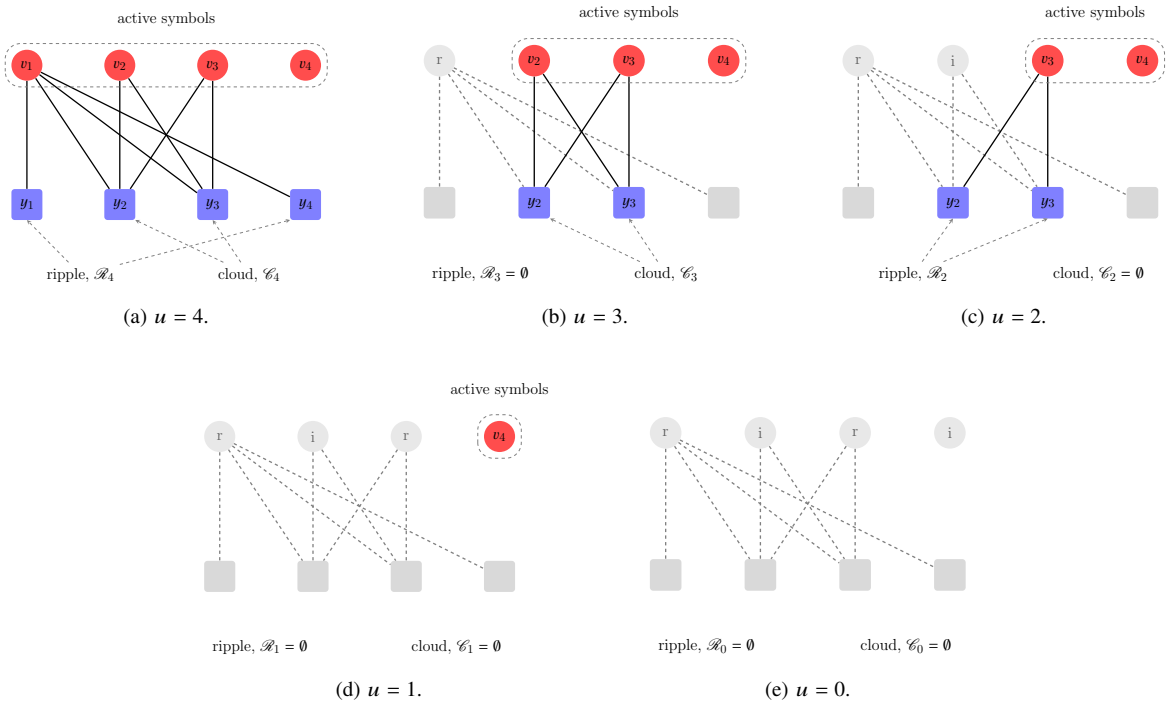
$$\mathsf{S}_u := (\mathtt{C}_u, \mathtt{R}_u).$$

We aim at deriving a recursion for the decoder state, that is, to obtain $\Pr\{\mathsf{S}_{u-1} = (\mathtt{c}_{u-1}, \mathtt{r}_{u-1})\}$ as a function of $\Pr\{\mathsf{S}_u = (\mathtt{c}_u, \mathtt{r}_u)\}$. We do so by analyzing how the ripple and cloud change in the transition from u to $u - 1$. In the transition exactly one active input symbol is marked as either resolvable or inactive and all its adjacent edges are removed. Whenever edges are removed from the graph, the reduced degree of one or more output symbols decreases. Consequently, some of symbols in the cloud may enter the ripple and some of the symbols in the ripple may become of reduced degree zero and leave the graph.

We focus first on the symbols that leave the cloud and enter the ripple during the transition at step u, conditioned on $\mathsf{S}_u = (\mathtt{c}_u, \mathtt{r}_u)$. Since for an LT code the neighbors of the output symbols are selected independently and uniformly at random, in the transition each output symbol may leave the cloud and enter the ripple independently from the other output symbols. Hence, the number of symbols leaving $\mathscr{C}_u$ and entering $\mathscr{R}_{u-1}$ is binomially distributed with parameters $\mathtt{c}_u$ and P_u with

$$\begin{aligned} \mathsf{P}_u :&= \Pr\{Y \in \mathscr{R}_{u-1} | Y \in \mathscr{C}_u\} \\ &= \frac{\Pr\{Y \in \mathscr{R}_{u-1},\, Y \in \mathscr{C}_u\}}{\Pr\{Y \in \mathscr{C}_u\}} \end{aligned} \tag{2}$$

[2]This is certainly neither the only possible inactivation strategy nor the one leading to the least number or inactivations. However, this strategy makes the analysis tractable. For an overview of the different inactivation strategies we refer the reader to [38].



(a) $u = 4$. (b) $u = 3$. (c) $u = 2$.

(d) $u = 1$. (e) $u = 0$.

Fig. 3. Triangulation procedure example

where random variable Y represents a randomly chosen output symbol.

We first consider the numerator of (2). Conditioning on the original degree of Y, we have the following proposition.

Proposition 1. *The probability that an output symbol Y belongs to the cloud at step u and enters the ripple at step $u-1$, condition to its original degree being d, is*

$$\Pr\{Y \in \mathscr{R}_{u-1}, Y \in \mathscr{C}_u | \deg(Y) = d\} = \begin{cases} (u-1)\binom{k-u}{d-2}\binom{k}{d}^{-1} & \text{if } 2 \le d \le k-u+2, \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

Proof: For an output symbol Y of degree d to belong to the cloud at step u and to the ripple at step $u-1$ we need that the output symbol has *reduced* degree 2 *before* the transition and *reduced* degree 1 *after* the transition. For this to happen two events must take place:

- $Y \in \mathscr{R}_{u-1}$ one of the d edges of output symbol Y is connected to the symbol being marked as inactive or resolvable at the transition,
- $Y \in \mathscr{C}_u$ another edge is connected to one of the $u-1$ active symbols after the transition and at the same time the remaining $d-2$ edges connected to the $k-u$ not active input symbols (inactive or resolvable).

The joint probability of $Y \in \mathscr{R}_{u-1}$ and $Y \in \mathscr{C}_u$ can be derived as

$$\Pr\{Y \in \mathscr{R}_{u-1}, Y \in \mathscr{C}_u\} = \Pr\{Y \in \mathscr{R}_{u-1}\} \Pr\{Y \in \mathscr{C}_u | Y \in \mathscr{R}_{u-1}\}$$

Let us focus first on $Y \in \mathscr{R}_{u-1}$. We consider an output symbols of degree d, thus this is simply probability that one the d edges of Y is connected to the symbol being marked as inactive or resolvable at the transition, $\Pr\{Y \in \mathscr{R}_{u-1}\} = d/k$.

Let us now focus on $\Pr\{Y \in \mathscr{C}_u | Y \in \mathscr{R}_{u-1}\}$. Since we condition on $Y \in \mathscr{R}_{u-1}$ we have that one of the d edges of Y is already assigned to one input symbol. We now need to consider the remaining $d-1$ edges and $k-1$ input symbols. The probability that one of the $d-1$ edges is connected to the set of $u-1$ active symbols is $(d-1)(u-1)/(k-1)$. At the same time we must have exactly $d-2$ edges going to the $k-u$ input symbols that were not active before the transition. Hence, we have

$$\Pr\{Y \in \mathscr{C}_u | Y \in \mathscr{R}_{u-1}\} = (d-1)\frac{u-1}{k-1}\binom{k-u}{d-2}\binom{k-2}{d-2}^{-1}$$

By multiplying the two probabilities, applying few manipulations, and making explicit the conditioning on $\deg(Y) = d$, we get

$$\Pr\{Y \in \mathscr{R}_{u-1}, Y \in \mathscr{C}_u | \deg(Y) = d\} = (u-1)\binom{k-u}{d-2}\binom{k}{d}^{-1}.$$

We shall anyhow observe that if $\deg(Y) < 2$, the output symbol cannot belong to the cloud. Moreover, one needs to impose the condition $d-2 \le k-u$, since output symbols choose their neighbors without replacement (an output symbol cannot have more than one edge going to an input symbol). Hence, for $d < 2$ and $d > k-u+2$, the probability $\Pr\{Y \in \mathscr{R}_{u-1}, Y \in \mathscr{C}_u | \deg(Y) = d\}$ is zero. ■

By removing the conditioning on the degree of Y in (3), we

have

$$\Pr\{Y \in \mathscr{R}_{u-1},\, Y \in \mathscr{C}_u\} = (u-1)\sum_{d=2}^{k-u+2} \Omega_d \binom{k-u}{d-2}\binom{k}{d}^{-1}.$$

Let us now focus on the denominator of (2) which gives the probability that a randomly chosen output symbol Y is in the cloud when u input symbols are still active. This probability is provided by the following proposition.

Proposition 2. *The probability that the randomly chosen output symbol Y is in the cloud when u input symbols are still active is*

$$\Pr\{Y \in \mathscr{C}_u\} = 1 - u\sum_{d=1}^{k-u+1} \Omega_d \binom{k-u}{d-1}\binom{k}{d}^{-1} + - \sum_{d=1}^{k-u} \Omega_d \binom{k-u}{d}\binom{k}{d}^{-1}. \tag{4}$$

Proof: The probability of Y not being in the cloud is given by the probability of Y having reduced degree 0 or being in the ripple. Given that the two events are mutually exclusive, we can compute such probability as the sum of the probabilities of the two events,

$$\begin{aligned}\Pr\{Y \in \mathscr{C}_u\} &= 1 - \Pr\{Y \in \mathscr{R}_u \cup \mathrm{red}_u(Y) = 0\} \\ &= 1 - \Pr\{Y \in \mathscr{R}_u\} - \Pr\{\mathrm{red}_u(Y) = 0\} \end{aligned} \tag{5}$$

where $\mathrm{red}_u(Y)$ denotes the reduced degree of output symbol Y when u input symbols are still active. We first focus on the probability of Y being in the ripple. Let us assume Y has original degree d. The probability that Y has reduced degree 1 equals the probability of Y having exactly one neighbor among the u active input symbols and the remaining $d-1$ neighbors among the $k-u$ non-active (i.e., solved or inactive) ones. Thus, we have

$$\Pr\{Y \in \mathscr{R}_u | \deg(Y) = d\} = d\frac{u}{k}\frac{\binom{k-u}{d-1}}{\binom{k-1}{d-1}} = u\binom{k-u}{d-1}\binom{k}{d}^{-1}. \tag{6}$$

The probability of Y having reduced degree 0 is the probability that all d neighbors of Y are in the $k-u$ non-active symbols. The total number of different edge assignments is $\binom{k}{d}$, and the total number of edge assignments in which all d neighbors of Y are in the $k-u$ non-active symbols is $\binom{k-u}{d}$. Thus we have

$$\Pr\{\mathrm{red}_u(Y) = 0 | \deg(Y) = d\} = \binom{k-u}{d}\binom{k}{d}^{-1}. \tag{7}$$

By removing the conditioning on d in (6) and (7) and by replacing the corresponding results in (5) we obtain (4). ∎

The expression of P_u is finally given in the following proposition.

Proposition 3. *The probability P_u that a randomly chosen output symbol leaves the cloud $\mathscr{C}_u$ enters the ripple $\mathscr{R}_{u-1}$ after the transition from u to $u-1$ active input symbols is*

$$\mathsf{P}_u = \frac{(u-1)\sum_{d=2}^{k-u+2} \Omega_d \binom{k-u}{d-2}\binom{k}{d}^{-1}}{1 - u\sum_{d=1}^{k-u+1} \Omega_d \binom{k-u}{d-1}\binom{k}{d}^{-1} - \sum_{d=1}^{k-u} \Omega_d \binom{k-u}{d}\binom{k}{d}^{-1}}.$$

Proof: The proof follows directly from (2) and Propositions 1 and 2. ∎

Let us now focus on the number $\mathtt{a}_u$ of symbols leaving the ripple during the transition from u to $u-1$ active symbols. We denote by $\mathtt{A}_u$ the random variable associated with $\mathtt{a}_u$. Two cases shall be considered. In a first case, no inactivation takes place because the ripple is not empty. Thus, an output symbol Y is chosen at random from the ripple and its only neighbor v is marked as resolvable and removed from the graph. By removing v from the graph, other output symbols that are connected to v and that are in the ripple leave the ripple. Thus, for $\mathtt{r}_u > 0$ we have

$$\Pr\{\mathtt{A}_u = \mathtt{a}_u | \mathtt{R}_u = \mathtt{r}_u\} = \binom{\mathtt{r}_u - 1}{\mathtt{a}_u - 1}\left(\frac{1}{u}\right)^{\mathtt{a}_u - 1}\left(1 - \frac{1}{u}\right)^{\mathtt{r}_u - \mathtt{a}_u} \tag{8}$$

with $1 \le \mathtt{a}_u \le \mathtt{r}_u$. In the second case, the ripple is empty ($\mathtt{r}_u = 0$) and an inactivation takes place. Since no output symbols can leave the ripple, we have

$$\Pr\{\mathtt{A}_u = \mathtt{a}_u | \mathtt{R}_u = 0\} = \begin{cases} 1 & \text{if } \mathtt{a}_u = 0 \\ 0 & \text{if } \mathtt{a}_u > 0. \end{cases} \tag{9}$$

We are now in the position to derive the transition probability $\Pr\{\mathsf{S}_{u-1} = (\mathtt{c}_{u-1}, \mathtt{r}_{u-1}) | \mathsf{S}_u = (\mathtt{c}_u, \mathtt{r}_u)\}$. Let us introduce the cloud drift $\mathtt{b}_u$ to denote the variation of number of cloud elements in the transition from u to $u-1$ active symbols, i.e.,

$$\mathtt{b}_u := \mathtt{c}_u - \mathtt{c}_{u-1}.$$

We can now express the cloud and ripple cardinality after the transition respectively as

$$\begin{aligned}\mathtt{c}_{u-1} &= \mathtt{c}_u - \mathtt{b}_u \\ \mathtt{r}_{u-1} &= \mathtt{r}_u - \mathtt{a}_u + \mathtt{b}_u.\end{aligned}$$

Since output symbols are generated independently, the random variable associated to $\mathtt{b}_u$ is binomially distributed with parameters $\mathtt{c}_u$ and P_u, and making use of (8), we obtain the following recursion

$$\begin{aligned}&\Pr\{\mathsf{S}_{u-1} = (\mathtt{c}_u - \mathtt{b}_u, \mathtt{r}_u - \mathtt{a}_u + \mathtt{b}_u) | \mathsf{S}_u = (\mathtt{c}_u, \mathtt{r}_u)\} \\ &= \binom{\mathtt{c}_u}{\mathtt{b}_u}\mathsf{P}_u^{\mathtt{b}_u}(1 - \mathsf{P}_u)^{\mathtt{c}_u - \mathtt{b}_u} \\ &\times \binom{\mathtt{r}_u - 1}{\mathtt{a}_u - 1}\left(\frac{1}{u}\right)^{\mathtt{a}_u - 1}\left(1 - \frac{1}{u}\right)^{\mathtt{r}_u - \mathtt{a}_u}\end{aligned} \tag{10}$$

which is valid for $\mathtt{r}_u > 0$, while for $\mathtt{r}_u = 0$, according to (9), we have

$$\Pr\{\mathsf{S}_{u-1} = (\mathtt{c}_u - \mathtt{b}_u, \mathtt{b}_u) | \mathsf{S}_u = (\mathtt{c}_u, 0)\} = \binom{\mathtt{c}_u}{\mathtt{b}_u} \mathsf{P}_u^{\mathtt{b}_u} (1 - \mathsf{P}_u)^{\mathtt{c}_u - \mathtt{b}_u}. \quad (11)$$

Note that the probability of $\mathsf{S}_{u-1} = (\mathtt{c}_{u-1}, \mathtt{r}_{u-1})$ can be computed recursively via (10), (11) by initializing the decoder state as

$$\Pr\{\mathsf{S}_k = (\mathtt{c}_k, \mathtt{r}_k)\} = \binom{m}{\mathtt{r}_k} \Omega_1^{\mathtt{r}_k} (1 - \Omega_1)^{\mathtt{c}_k}$$

for all non-negative $\mathtt{c}_k, \mathtt{r}_k$ such that $\mathtt{c}_k + \mathtt{r}_k = m$, where m is the number of output symbols.

The following proposition establishes how the number of inactivations can be determined using the finite state machine.

Theorem 1. *Let* $\mathtt{T}$ *denote the random variable associated to the number of inactivations at the end of the triangulation process. The expected value of* $\mathtt{T}$ *is given by*

$$\mathsf{E}[\mathtt{T}] = \sum_{u=1}^{k} \sum_{\mathtt{c}_u} \Pr\{\mathsf{S}_u = (\mathtt{c}_u, 0)\}. \quad (12)$$

Proof: Let us denote by $\mathbf{f} = (\mathtt{f}_1, \mathtt{f}_2, \cdots, \mathtt{f}_k)$ the binary vector associated to the inactivations performed during the triangulation process, and let $\mathbf{F} = (\mathtt{F}_1, \mathtt{F}_2, \cdots, \mathtt{F}_k)$ denote the associated random vector. In particular, the u-th element of $\mathbf{f}$, $\mathtt{f}_u$ is set to 1 if an inactivation is performed when u input symbols are active, and it is set to 0 if no inactivation is performed. Thus, for a given instance of inactivation decoding, the total number of inactivations corresponds simply to the Hamming weight of $\mathbf{f}$, which we denote as $w_H(\mathbf{f})$. The expected number of inactivations can be obtained as

$$\begin{aligned}\mathsf{E}[\mathtt{T}] &= \sum_{\mathbf{f}} w_H(\mathbf{f}) \Pr\{\mathbf{F} = \mathbf{f}\} \\ &= \sum_{\mathbf{f}} \left(\sum_u \mathtt{f}_u \right) \Pr\{\mathbf{F} = \mathbf{f}\} = \sum_u \sum_{\mathbf{f}} \mathtt{f}_u \Pr\{\mathbf{F} = \mathbf{f}\}\end{aligned}$$

where the summation is taken over all possible vectors $\mathbf{f}$. We shall now define $\mathbf{f}_{\setminus u} = (\mathtt{f}_1, \cdots, \mathtt{f}_{u-1}, \mathtt{f}_{u+1}, \mathtt{f}_k)$, i.e., $\mathbf{f}_{\setminus u}$ denotes a vector containing all but the u-th element of $\mathbf{f}$. We have

$$\begin{aligned}\mathsf{E}[\mathtt{T}] &= \sum_u \sum_{\mathbf{f}_{\setminus u}} \sum_{\mathtt{f}_u} \mathtt{f}_u \Pr\{\mathbf{F} = \mathbf{f}\} \\ &= \sum_u \sum_{\mathtt{f}_u} \mathtt{f}_u \sum_{\mathbf{f}_{\setminus u}} \Pr\{\mathbf{F} = \mathbf{f}\} \\ &= \sum_u \sum_{\mathtt{f}_u} \mathtt{f}_u \Pr\{\mathtt{F}_u = \mathtt{f}_u\} \\ &= \sum_u \Pr\{\mathtt{F}_u = 1\}.\end{aligned}$$

If we now observe that by definition

$$\Pr\{\mathtt{F}_u = 1\} = \sum_{\mathtt{c}_u} \Pr\{\mathsf{S}_u = (\mathtt{c}_u, 0)\}$$

we obtain (12), and the proof is complete. ∎

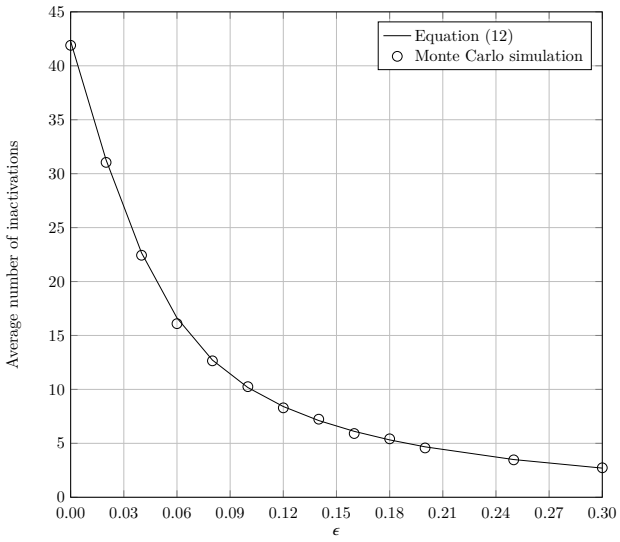


Fig. 4. Average number of inactivations vs. relative overhead ϵ for an LT code with $k = 1000$ and with degree distribution $\Omega^{(1)}(\mathtt{x})$.

Figure 4 shows the expected number of inactivations for a $k = 1000$ LT code with the output degree distribution

$$\begin{aligned}\Omega^{(1)}(\mathtt{x}) :&= 0.0098\mathtt{x} + 0.4590\mathtt{x}^2 + 0.2110\mathtt{x}^3 + 0.1134\mathtt{x}^4 + \\ &+ 0.1113\mathtt{x}^{10} + 0.0799\mathtt{x}^{11} + 0.0156\mathtt{x}^{40} \quad (13)\end{aligned}$$

which is the one adopted by standardized Raptor codes [17], [39]. The figure shows the number of inactivations according to (12) and results obtained through Monte Carlo simulations. In particular, in order to obtain the simulation results for each value of relative overhead 1000 decoding attempts were carried out. A tight match between the analysis and the simulation results can be observed.

The results in this section are strongly based on [16], where LT codes under inactivation decoding were analyzed. The difference is that, when considering the triangulation step of inactivation decoding, the decoding process does not stop when the decoder is at state $\mathsf{S}_u = (\mathtt{c}_u, 0)$. Instead, decoding can be resumed after performing an inactivation.

B. Distribution of the Number of Inactivations

The analysis presented in Section IV-A provides the expected number of inactivations at the end of the triangulation process under random inactivation decoding. In this section we extend the analysis to obtain the probability distribution of the number of inactivations. To do so, we extend the finite state machine by including the number of inactive input symbols in the state definition, i.e.,

$$\mathsf{S}_u = (\mathtt{C}_u, \mathtt{R}_u, \mathtt{T}_u)$$

where $\mathtt{T}_u$ is the random variable associated to the number of inactivations at step u (when u input symbols are active). We proceed by deriving a recursion to obtain $\Pr\{\mathsf{S}_{u-1} = (\mathtt{c}_{u-1}, \mathtt{r}_{u-1}, \mathtt{t}_{u-1})\}$ as a function of $\Pr\{\mathsf{S}_u = (\mathtt{c}_u, \mathtt{r}_u, \mathtt{t}_u)\}$. Two cases shall be considered. When the ripple is not empty ($\mathtt{r}_u > 0$) no inactivation takes place,

at the transition from u to $u-1$ active symbols the number of inactivations is unchanged ($\mathtt{t}_{u-1} = \mathtt{t}_u$). Hence, we have

$$\begin{aligned}\Pr\{\mathsf{S}_{u-1} &= (\mathtt{c}_u - \mathtt{b}_u, \mathtt{r}_u - \mathtt{a}_u + \mathtt{b}_u, \mathtt{t}_u) | \mathsf{S}_u = (\mathtt{c}_u, \mathtt{r}_u, \mathtt{t}_u)\} \\ &= \binom{\mathtt{c}_u}{\mathtt{b}_u} \mathsf{P}_u{}^{\mathtt{b}_u} (1-\mathsf{P}_u)^{\mathtt{c}_u - \mathtt{b}_u} \\ &\times \binom{\mathtt{r}_u - 1}{\mathtt{a}_u - 1} \left(\frac{1}{u}\right)^{\mathtt{a}_u - 1} \left(1 - \frac{1}{u}\right)^{\mathtt{r}_u - \mathtt{a}_u}. \end{aligned} \tag{14}$$

If the ripple is empty ($\mathtt{r}_u = 0$) an inactivation takes place. In this case the number of inactivations increases by one yielding

$$\begin{aligned}\Pr\{\mathsf{S}_{u-1} &= (\mathtt{c}_u - \mathtt{b}_u, \mathtt{b}_u, \mathtt{t}_u + 1) | \mathsf{S}_u = (\mathtt{c}_u, 0, \mathtt{t}_u)\} \\ &= \binom{\mathtt{c}_u}{\mathtt{b}_u} \mathsf{P}_u{}^{\mathtt{b}_u} (1-\mathsf{P}_u)^{\mathtt{c}_u - \mathtt{b}_u}. \end{aligned} \tag{15}$$

The probability of $\mathsf{S}_{u-1} = (\mathtt{c}_{u-1}, \mathtt{r}_{u-1}, \mathtt{t}_{u-1})$ can be computed recursively via (14), (15) starting with the initial condition

$$\Pr\{\mathsf{S}_k = (\mathtt{c}_k, \mathtt{r}_k, \mathtt{t}_u)\} = \binom{m}{\mathtt{r}} \Omega_1^{\mathtt{r}} (1-\Omega_1)^{\mathtt{c}_k}$$

for all non-negative $\mathtt{c}_k, \mathtt{r}_k$ such that $\mathtt{c}_k + \mathtt{r}_k = m$ and $\mathtt{t}_k = 0$. Finally, the distribution of the number of inactivations needed to complete the decoding process is given by[3]

$$f_{\mathrm{T}}(\mathtt{t}) = \sum_{\mathtt{c}_0} \sum_{\mathtt{r}_0} \Pr\{\mathsf{S}_0 = (\mathtt{c}_0, \mathtt{r}_0, \mathtt{t})\}. \tag{16}$$

In Figure 5 the distribution of the number of inactivations is shown, for an LT code with degree distribution $\Omega^{(1)}(\mathtt{x})$ given in (13), input block size $k = 300$ and relative overhead $\epsilon = 0.02$. The chart shows the distribution of the number of inactivations obtained through Monte Carlo simulations and by evaluating (16). In order to obtain the simulation results for each value of absolute overhead 10000 decoding attempts were carried out. As before, we can observe a very tight match between the analysis and the simulation results.

V. Inactivation Decoding of Raptor Codes

The analysis introduced in Section IV holds for LT codes. We shall see next how the proposed tools can be successfully applied to the design of Raptor codes whose outer code has a dense parity check matrix (see [30]). We proceed by illustrating the impact of the LT component of a Raptor code on the inactivation count. We then develop a methodology for the design of Raptor codes based on the results of Section IV.

[3]From (16) we may obtain the cumulative distribution $F_{\mathrm{T}}(\mathtt{t})$. The cumulative distribution of the number of inactivations has practical implications. Let us assume the fountain decoder runs on a platform with limited computational capability. For example, the decoder may be able to handle a maximum number of inactive symbols (recall that the complexity of inactivation decoding is cubic in the number of inactivations, $\mathtt{t}$). Suppose the maximum number of inactivations that the decoder can handle is $\mathtt{t}_{\max}$. The probability of decoding failure will be lower bounded by $1 - F_{\mathrm{T}}(\mathtt{t}_{\max})$.The probability of decoding failure is actually higher than $1 - F_{\mathrm{T}}(\mathtt{t}_{\max})$ since the system of equations to be solved in the Gaussian elimination (GE) step might be rank deficient.

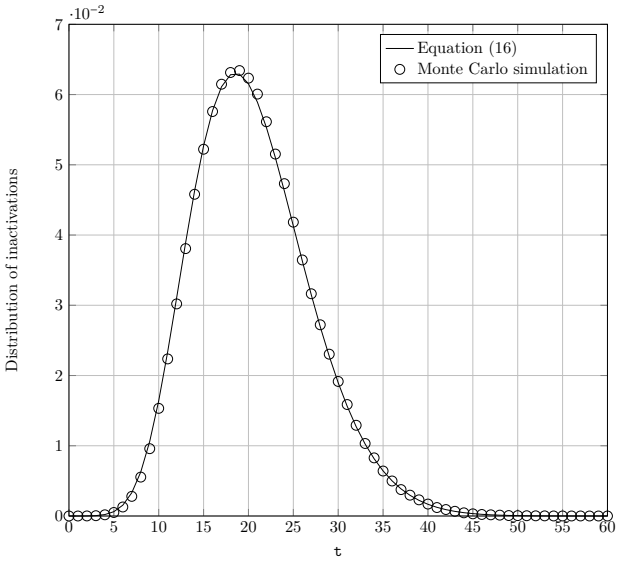


Fig. 5. Distribution of the number of inactivations for an LT code with $k = 300$, relative overhead $\epsilon = 0.02$ and degree distribution $\Omega^{(1)}(\mathtt{x})$ given in (13).

A. Average Number of Inactivations for Raptor Codes

Let us now consider a Raptor code based on the concatenation of a (dense) (h, k) outer code and an inner LT code. We denote by $\boldsymbol{H}_{\mathrm{P}}$ the $((h-k) \times h)$ parity-check matrix of the outer code. At the input of the Raptor encoder, we have a vector of k input symbols, $\boldsymbol{u} = (u_1, u_2, \ldots, u_k)$. Out of the input symbols, the outer encoder generates a vector of h intermediate symbols $\boldsymbol{v} = (v_1, v_2, \ldots, v_h)$. The intermediate symbols serve as input to an LT encoder which produces the codeword $\boldsymbol{c} = (c_1, c_2, \ldots, c_n)$, where n can grow unbounded. The relation between intermediate and output symbols can be expressed as $\boldsymbol{c} = \boldsymbol{v}\boldsymbol{G}$ where $\boldsymbol{G}$ is the generator matrix of the LT code. The output symbols are sent over a BEC, at the output of which the receiver collects $m = k + \delta$ output symbols, denoted as $\boldsymbol{y} = (y_1, y_2, \ldots, y_m)$. The relation between the collected output symbols and the intermediate symbols can be expressed as

$$\boldsymbol{G}_0^{\mathsf{T}} \boldsymbol{v}^{\mathsf{T}} = \boldsymbol{y}^{\mathsf{T}}.$$

where $\boldsymbol{G}_0$ is a matrix that contains the m columns of $\boldsymbol{G}$ associated to the m received output symbols. Due to the outer code constraints, we have

$$\boldsymbol{H}_{\mathrm{P}} \boldsymbol{v}^{\mathsf{T}} = \boldsymbol{0}^{\mathsf{T}}$$

where $\boldsymbol{0}$ is the length-$(h-k)$ zero vector. Let us now define the constraint matrix $\boldsymbol{M}$ of the Raptor code

$$\boldsymbol{M} := \left[\boldsymbol{H}_{\mathrm{P}}^{\mathsf{T}} | \boldsymbol{G}_0\right].$$

ML decoding can be carried out by solving the system

$$\boldsymbol{M}^{\mathsf{T}} \boldsymbol{v}^{\mathsf{T}} = \left[\boldsymbol{0} | \boldsymbol{y}\right]^{\mathsf{T}}. \tag{17}$$

If we compare the system of equations that need to be solved for LT and Raptor decoding, given respectively by (1) and (17), we can observed how Raptor ML decoding is very similar to

ML decoding of an LT code. The main difference lies in the fact that matrix $\boldsymbol{M}$ is formed, for the first $h-k$ columns, by the transpose of the outer code parity-check matrix. The high density of the outer code parity-check matrix lowers the probability that the ripple contains (some of) the $h-k$ output symbols associated to the zero vector in (17) (this is especially evident at the early steps of the triangulation process). As a result, we shall expect the average number of inactivations to increase with $h-k$, for a fixed overhead [4].

In Figure 6 we provide the average number of inactivations needed to decode two Raptor codes, as a function of the receiver overhead δ. Both Raptor codes have the same outer code, a $(63, 57)$ Hamming code, but different LT degree distributions. The first distribution is $\Omega^{(1)}(\mathrm{x})$ from (13), and the second distribution is

$$\Omega^{(2)}(\mathrm{x}) := 0.05\mathrm{x} + 0.2\mathrm{x}^2 + 0.4\mathrm{x}^3 + 0.3\mathrm{x}^4 + 0.05\mathrm{x}^{40}.$$

The Figure also shows the number of inactivations needed to decode the two standalone LT codes. If we compare the number of inactivations required by the Raptor and LT codes, we can see how for both degree distributions, the number of additional inactivations needed for Raptor decoding with respect to LT decoding is very similar. We hence conjecture that the impact of the (dense) outer code on the inactivation count depends mostly on the number of the outer code parity-check equations. This empirical observation provides a hint on a practical design strategy for Raptor codes: If one aims at minimizing the number of inactivations for a Raptor code based on a given outer code, it is sufficient to design the LT code component for a low (i.e., minimal) number of inactivations. Based on this consideration, an explicit design example is provided in the following subsection.

B. Example of Raptor Code Design

We consider a Raptor code with a $(63, 57)$ outer Hamming code and we assume decoding is carried out when the absolute receiver overhead reaches $\delta = \delta^*$, i.e., we carry out decoding after collecting δ^* output symbols in excess of k. Furthermore, we want to have a probability of decoding failure, $\mathsf{P_F}$, lower that a given value $\mathsf{P_F}^\star$. Thus, the objective of our code design will be minimizing the number of inactivations needed for decoding while achieving a probability of decoding failure lower than $\mathsf{P_F}^\star$. Hence, the design problem consists of finding a suitable output degree distribution for the inner LT code. For illustration we will introduce a series of constraints in the output degree distribution. In particular we constraint the output degree distribution to have the same maximum and average output degree as standard R10 Raptor codes ($\bar{\Omega} \approx 4.63$ and $d_{\max} = 40$, [17]). Furthermore, we constraint the output degree distribution to have the same support as the degree distribution of R10 raptor codes, that is, only degrees $1, 2, 3, 4, 10, 11$ and 40 are allowed. These design constraints allow us to perform a fair comparison between the Raptor code obtained through optimization and a Raptor code with the same outer code and the degree distribution from R10 Raptor codes.

[4]In practice, the outer codes used are not totally dense, but the rows of their parity check matrix are usually denser that the rows of $\boldsymbol{G}^{\mathsf{T}}$. This is, for example, usually the case if the outer code is a high rate LDPC code, since the check node degree increases with the rate.

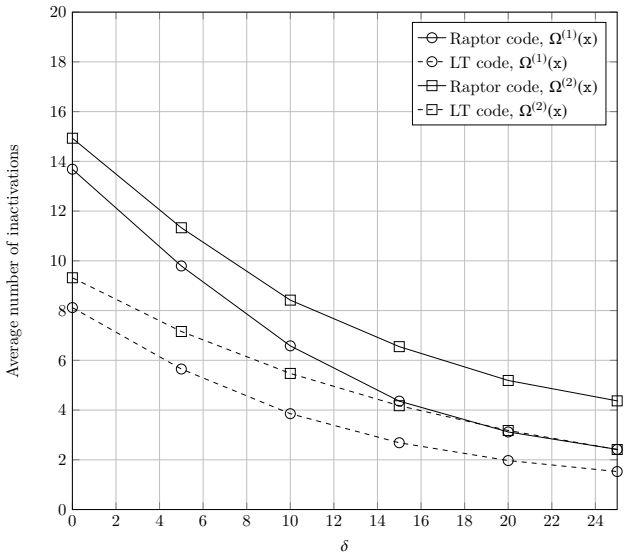


Fig. 6. Expected number of inactivations two Raptor codes using a $(63, 57)$ Hamming code with LT degree distributions $\Omega^{(1)}(\mathrm{x})$ and $\Omega^{(2)}(\mathrm{x})$, and for two LT codes with $k = 63$ and degree distributions $\Omega^{(1)}(\mathrm{x})$ and $\Omega^{(2)}(\mathrm{x})$.

The design of the LT output degree distribution is formulated as a numerical optimization problem. For the numerical optimization we used is simulated annealing (SA) [40]. More concretely, we define the objective function to be minimized to be the following function [41]

$$\eta = \mathsf{E}[\mathtt{T}] + \phi\left(\bar{\mathsf{P}}_\mathsf{F}\right)$$

where $\mathsf{E}[\mathtt{T}]$ is the number of inactivations needed to decode the LT code and the penalty function ϕ is defined as

$$\phi\left(\bar{\mathsf{P}}_\mathsf{F}\right) = \begin{cases} 0 & \text{if } \bar{\mathsf{P}}_\mathsf{F} < \mathsf{P_F}^\star \\ b\ \left(1 - \mathsf{P_F}^\star / \bar{\mathsf{P}}_\mathsf{F}\right) & \text{otherwise} \end{cases}$$

being b a large positive constant[5], $\mathsf{P_F}^\star$ the target probability of decoding failure at $\delta = \delta^*$ and $\bar{\mathsf{P}}_\mathsf{F}$ the upper bound to the probability of decoding failure of the Raptor code in [33], which for binary Raptor codes has the expression[6]

$$\mathsf{P_F} \leq \bar{\mathsf{P}}_\mathsf{F} := \sum_{l=1}^{h} A_l^{\mathrm{o}} \pi_l^{k+\delta}$$

where A_l^{o} is the multiplicity of codewords of weight l in the outer code, and π_l depends on the LT code output degree

[5]In our example, b was set to 10^4. A large b factor ensures that degree distributions which do not comply with the target probability of decoding failure are discarded.

[6]The use of the upper bound on the probability of decoding failure, $\bar{\mathsf{P}}_\mathsf{F}$ in place of the actual value of $\mathsf{P_F}$ in the objective functions stems from the need of having a fast (though, approximate) performance estimation to be used within the SA recursion. The evaluation of the actual $\mathsf{P_F}$ presents a prohibitive complexity since it has to be obtained through Monte Carlo simulations. Note also that the upper bound of [33] is very tight.

distribution as

$$\pi_l = \sum_{j=1}^{d_{\max}} \Omega_j \sum_{\substack{i=\max(0,l+j-h) \\ i \text{ even}}}^{\min(l,j)} \frac{\binom{j}{i}\binom{h-j}{l-i}}{\binom{h}{l}}.$$

In particular, two code designs were carried out using the proposed optimization. In the first case we set the overhead to $\delta^* = 15$ and the target probability of decoding failure to $\mathsf{P_F}^\star = 10^{-3}$, and we denote by $\Omega^{(3)}$ the distribution obtained from the optimization process. In the second case we chose the same overhead, $\delta^* = 15$, and set the target probability of decoding failure to $\mathsf{P_F}^\star = 10^{-2}$, and we denote by $\Omega^{(4)}$ the resulting distribution. The degree distributions obtained are the following

$$\Omega^{(3)}(\mathrm{x}) = 0.0347\ \mathrm{x}^1 + 0.3338\ \mathrm{x}^2 + 0.2268\ \mathrm{x}^3 + 0.1548\ \mathrm{x}^4 + 0.1515\ \mathrm{x}^{10} + 0.0973\ \mathrm{x}^{11} + 0.0011\ \mathrm{x}^{40}$$

$$\Omega^{(4)}(\mathrm{x}) = 0.0823\ \mathrm{x}^1 + 0.4141\ \mathrm{x}^2 + 0.1957\ \mathrm{x}^3 + 0.1272\ \mathrm{x}^4 + 0.0797\ \mathrm{x}^{10} + 0.0762\ \mathrm{x}^{11} + 0.0248\ \mathrm{x}^{40}.$$

Monte Carlo simulations were carried out in order to assess the performance of the two Raptor codes obtained as result of the optimization process. In order to have a benchmark for comparison, a third Raptor code was considered, employing the same outer code (Hamming) and the degree distribution of standard R10 Raptor codes given in (13). Note that in all three cases we consider the same outer code, a $(63, 57)$ Hamming code, and thus, the number of input symbols is $k = 57$. To derive the probability of decoding failure for each overhead value δ simulations were run until 200 errors were collected, whereas in order to obtain the average number of inactivations, 1000 transmissions were simulated for each overhead value δ.

Figure 7 shows the probability of decoding failure $\mathsf{P_F}$ as a function of δ for the three Raptor codes based on the $(63, 57)$ outer Hamming code and inner LT codes with degree distributions $\Omega^{(1)}(\mathrm{x})$, $\Omega^{(3)}(\mathrm{x})$ and $\Omega^{(4)}(\mathrm{x})$. The upper bound to the probability of failure $\bar{\mathsf{P}}_\mathsf{F}$ is also provided. It can be observed how the Raptor codes with degree distributions $\Omega^{(3)}(\mathrm{x})$ and $\Omega^{(4)}(\mathrm{x})$ meet the design goal, being their probability of decoding failure at $\delta = 15$ below 10^{-3} and 10^{-2}, respectively. It can also be observed that the probability of decoding failure of the Raptor code with degree distribution $\Omega^{(1)}(\mathrm{x})$ lies between that of $\Omega^{(3)}(\mathrm{x})$ and $\Omega^{(4)}(\mathrm{x})$.

Figure 8 shows the average number of inactivations as a function of the absolute receiver overhead for the three Raptor codes considered. It can observed that the Raptor code requiring the least number of inactivations is $\Omega^{(4)}$, followed by $\Omega^{(1)}$, and finally the Raptor code with degree distribution $\Omega^{(3)}$ is the one requiring the most inactivations, and thus has the highest decoding complexity.

The results in Figures 7 and 8 illustrate the tradeoff existing between probability of decoding failure and number of inactivations (decoding complexity): In general if one desires to improve the probability of decoding failure, it is necessary to adopt LT codes with degree distributions that lead to a larger average number of inactivations.

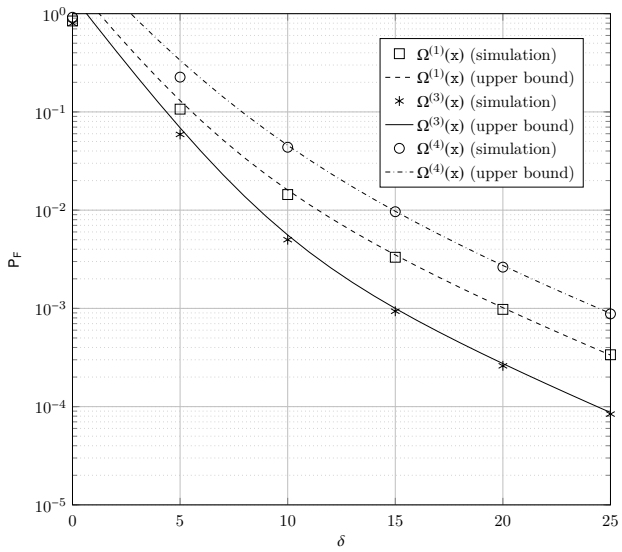


Fig. 7. Probability of decoding failure $\mathsf{P_F}$ vs. absolute receiver overhead δ for binary Raptor codes with a $(63, 57)$ Hamming outer code and LT degree distributions $\Omega^{(1)}(\mathrm{x})$, $\Omega^{(3)}(\mathrm{x})$ and $\Omega^{(4)}(\mathrm{x})$. The markers represent the result of simulations, while the lines represent the upper bound to the probability of decoding failure in [33].

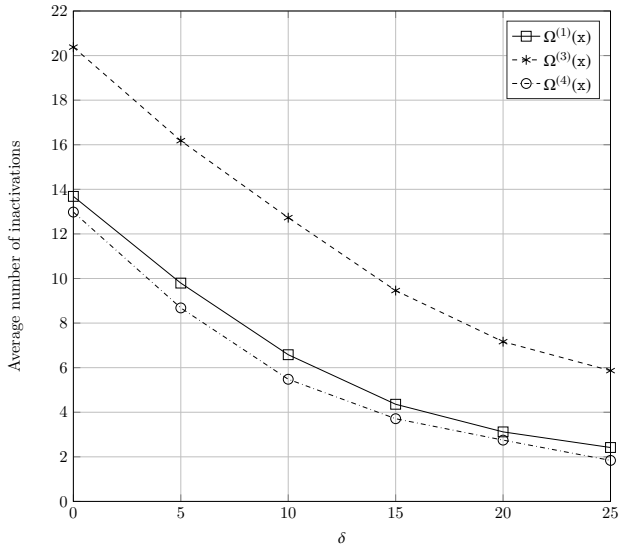


Fig. 8. Number of inactivations vs. absolute receiver overhead δ for binary Raptor codes with a $(63, 57)$ Hamming outer code and LT degree distributions $\Omega^{(1)}(\mathrm{x})$, $\Omega^{(3)}(\mathrm{x})$ and $\Omega^{(4)}(\mathrm{x})$.

VI. Conclusions

In this paper the decoding complexity of LT and Raptor codes under inactivation decoding has been analyzed. Using a dynamic programming approach, recursions for the number of inactivations have been derived for LT codes as a function of their degree distribution. The analysis has been extended to obtain the probability distribution of the number of inactivations. Furthermore, the experimental observation is made that decoding a Raptor code with a dense outer code results in an increase of the number of inactivations, compared to decoding a standalone LT code. Based on this observation, it has been

shown how the recursive analysis of LT codes can be used to design Raptor codes with a fine control on the number of inactivations vs. decoding failure probability trade-off.

APPENDIX A
INDEPENDENT POISSON APPROXIMATION

In Section IV we have derived recursive methods that can compute the expected number of inactivations and the distribution of the number of inactivations required by inactivation decoding. The proposed recursive methods, albeit accurate, entail a non negligible computational complexity. In this appendix we propose an approximate recursive method that can provide a reasonably-accurate estimation of the number of inactivations with a much lower computational burden.

The development of the approximate analysis relies on the following definition.

Definition 3 (Reduced degree-d set)**.** *The reduced degree-d set is the set of output symbols of reduced degree d. We denote it by $\mathscr{Z}_d$.*

The cardinality of $\mathscr{Z}_d$ is denoted by $\mathtt{z}_d$ and its associated random variable by $\mathtt{Z}_d$. Obviously, $\mathscr{Z}_1$ corresponds to the ripple. Furthermore, it is easy to see how the cloud $\mathscr{C}$ corresponds to the union of the sets of output symbols of reduced degree higher than 1, i.e.,

$$\mathscr{C} = \bigcup_{d=2}^{d_{\max}} \mathscr{Z}_d.$$

Moreover, since the sets $\mathscr{Z}_d$ are disjoint, we have

$$\mathtt{C} = \sum_{d=2}^{d_{\max}} \mathtt{Z}_d.$$

We aim at approximating the evolution of the number of reduced degree d output symbols, $\mathtt{Z}_d$, as the triangulation procedure of inactivation decoding progresses. As it was done in Section IV, in the following a temporal dimension shall be added through subscript u (recall that the subscript u corresponds to the number of active input symbols in the graph). At the beginning of the triangulation process we have $u = k$, and the counter u decreases by one in each step of triangulation. Triangulation ends when $u = 0$. It follows that $\mathscr{Z}_{d,u}$ is the set of reduced degree d output symbols when u input symbols are still active. Moreover, $\mathtt{Z}_{d,u}$ and $\mathtt{z}_{d,u}$ are respectively the random variable associated to the number of reduced degree d output symbols when u input symbols are still active and its realization. We model the triangulation process by means of a finite state machine with state

$$\mathsf{S}_u := \left(\mathtt{Z}_{1,u}, \mathtt{Z}_{2,u}, \ldots, \mathtt{Z}_{d_{\max},u}\right).$$

This model is equivalent to the one presented in Section IV-A. However, the evaluation of the state evolution is now more complex due to the large state space. Yet, the analysis can be, greatly simplified by resorting to an approximation. Before decoding starts (for $u = k$), due to the independence of output symbols we have that S_k follows a multinomial distribution, which for large number of output symbols m can be approximated as the product of independent Poisson distributions. Figure 9 shows the probability distribution of $\mathtt{Z}_{1,u}$, $\mathtt{Z}_{2,u}$ and $\mathtt{Z}_{3,u}$ for an LT code with a robust soliton distribution and $k = 1000$ obtained through Monte Carlo simulation, for $u = 1000$, 500 and 20. The figure also shows the curves of the Poisson distributions which match best the experimental data in terms of minimum mean square error. As we can observe, the Poisson distribution tightly matches the experimental data not only for $u = k$, but also for smaller values of u. Hence, we shall approximate the distribution of the decoder state at step u as a product of independent Poisson distributions,

$$\Pr\{\mathsf{S}_u = \mathtt{z}_u\} \approx \prod_{d=1}^{d_{\max}} \frac{\lambda_{d,u}{}^{\mathtt{z}_{d,u}} e^{-\lambda_{d,u}}}{\mathtt{z}_{d,u}!} \tag{18}$$

where $\mathbf{z}_u$ is the vector defined as $\mathbf{z}_u = \left(\mathtt{z}_{1,u}, \mathtt{z}_{2,u}, \ldots \mathtt{z}_{d_{\max},u},\right)$, i.e., we assume that the distribution of reduced degree d output symbols when u input symbols are active follows a Poisson distribution with parameter $\lambda_{d,u}$. We remark that by introducing this assumption, the number of received output symbols m is no longer constant but becomes a sum of Poisson random variables. In spite of this mismatch, we will later see how a good estimate of the number of inactivations can be obtained resorting to this approximation.

Next, we shall explain how the parameters $\lambda_{d,u}$ can be determined. For this purpose let us define $B_{d,u}$ as the random variable associated with the number of output symbols of reduced degree d that become of reduced degree $d-1$ in the transition from u to $u-1$ output symbols. We have

$$\mathtt{Z}_{d,u-1} = \mathtt{Z}_{d,u} + B_{d+1,u} - B_{d,u}.$$

If we take the expectation at both sides we can write

$$\mathsf{E}\left[\mathtt{Z}_{d,u-1}\right] = \mathsf{E}\left[\mathtt{Z}_{d,u}\right] + \mathsf{E}\left[B_{d+1,u}\right] - \mathsf{E}\left[B_{d,u}\right]. \tag{19}$$

Let us now derive the expression of $\mathsf{E}\left[B_{d,u}\right]$. We shall distinguish two cases. First we shall consider output symbols with reduced degree $d \geq 2$. Since output symbols select their neighbors uniformly at random, we have that $B_{d,u}$, $d \geq 2$, conditioned on $\mathtt{Z}_{d,u} = \mathtt{z}_{d,u}$ is binomially distributed with parameters $\mathtt{z}_{d,u}$ and $\mathsf{P}_{d,u}$, where $\mathsf{P}_{d,u}$ is the probability that the degree of Y decreases to $d-1$ in the transition from u to $u-1$, i.e.,

$$\mathsf{P}_{d,u} := \Pr\{Y \in \mathscr{Z}_{d-1,u-1} | Y \in \mathscr{Z}_{d,u}\}.$$

The following proposition holds.

Proposition 4. *The probability that a randomly chosen output symbol Y, with reduced degree $d \geq 2$ when u input symbols are active, has reduced degree $d-1$ when $u-1$ input symbols are active is*

$$\mathsf{P}_{d,u} = \frac{d}{u}.$$

Proof: Before the transition, Y has exactly d neighbors among the u active input symbols. In the transition from u to $u-1$ active symbols, 1 input symbol is selected at random and marked as either resolvable or inactive. The probability that

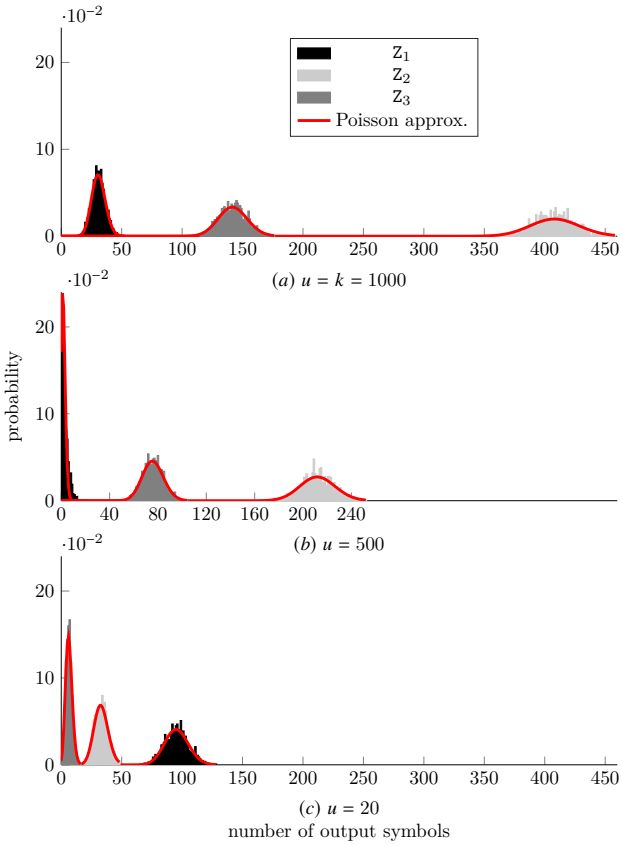


Fig. 9. Distribution of $Z_{1,u}$, $Z_{2,u}$ and $Z_{3,u}$ for an LT code with robust soliton distribution and $k = 1000$ obtained through Monte Carlo simulation. The upper, middle and lower figures represent respectively the distribution for $u = 1000$, 500 and 20. The black bars represent $Z_{1,u}$, the light grey bars $Z_{2,u}$ and the dark grey bars $Z_{3,u}$. The red lines represent the best Poisson distribution fit in terms of minimum mean square error.

the degree of Y gets reduced is simply the probability that one of its d neighbors is marked as resolvable or inactive. ■

Thus, the expected value of $B_{d,u}$ is

$$\mathsf{E}\left[B_{d,u}\right] = \mathsf{E}\left[\mathsf{E}\left[B_{d,u}|Z_{d,u}\right]\right] = \frac{d}{u}\mathsf{E}\left[Z_{d,u}\right]. \tag{20}$$

If we now replace (20) in (19) a recursive expression is obtained for $\lambda_{d,u}$, $d \geq 2$,

$$\lambda_{d,u-1} = \lambda_{d,u} + \frac{d+1}{u}\lambda_{d+1,u} - \frac{d}{u}\lambda_{d,u} \tag{21}$$
$$\lambda_{d,u-1} = \left(1 - \frac{d}{u}\right)\lambda_{d,u} + \frac{d+1}{u}\lambda_{d+1,u}$$

where we have replaced $\mathsf{E}\left[Z_{d,u}\right] = \lambda_{d,u}$ according to our Poisson distribution assumption.

We shall now consider the output symbols of reduced degree 1. In particular, we are interested in $B_{1,u}$, the random variable associated to the output symbols of reduced degree $d = 1$ that become of reduced degree 0 in the transition from u to $u-1$ active input symbols. Two different cases need to be considered. In the first one, the ripple is not empty, and hence there are one or more output symbols of reduced degree 1. In this case, an output symbol Y is chosen at random from the ripple and its only neighbor v is marked as resolvable and removed from the graph. Furthermore, any other output symbol in the ripple being connected to input symbol v also leaves the ripple during the transition. Hence, for $z_{1,u} \geq 1$ we have

$$\mathsf{E}\left[B_{1,u}|Z_{1,u} = z_{1,u} \geq 1\right] = 1 + \frac{1}{u}\left(z_{1,u} - 1\right)$$
$$= 1 - \frac{1}{u} + \frac{1}{u}z_{1,u}$$

whereas for $z_{1,u} = 0$ we have

$$\mathsf{E}\left[B_{1,u}|Z_{1,u} = 0\right] = 0.$$

Hence, we have

$$\mathsf{E}\left[B_{1,u}\right] = \left(1 - \frac{1}{u}\right)\Pr\{Z_{1,u} \geq 1\} + \frac{1}{u}\sum_{z_{1,u}=1}^{m} z_{1,u}\Pr\{Z_{1,u} = z_{1,u}\}$$

$$= \left(1 - \frac{1}{u}\right)\left(1 - \Pr\{Z_{1,u} = 0\}\right) + \frac{1}{u}\mathsf{E}\left[Z_{1,u}\right]$$
$$= \left(1 - \frac{1}{u}\right)\left(1 - e^{-\lambda_{1,u}}\right) + \frac{1}{u}\lambda_{1,u}. \tag{22}$$

Replacing (22) in (19) a recursive expression is obtained as,

$$\lambda_{1,u-1} = \left(1 - \frac{1}{u}\right)\lambda_{1,u} + \frac{2}{u}\lambda_{2,u} - \left(1 - \frac{1}{u}\right)\left(1 - e^{-\lambda_{1,u}}\right). \tag{23}$$

The decoder state probability is obtained by setting the initial condition $\lambda_{d,k} = m\,\Omega_d$ and applying the recursions in (21) and (23). Furthermore, the expected number of inactivations after the k steps of triangulation, can be approximated as

$$\mathsf{E}\left[I\right] = \sum_{u=1}^{k}\Pr\{Z_{1,u} = 0\} \approx \sum_{u=1}^{k} e^{-\lambda_{1,u}}.$$

In Figure 10 we provide again the probability distribution of $Z_{1,u}$, $Z_{2,u}$ and $Z_{3,u}$ for an LT code with robust soliton distribution (RSD) (see [5]) and $k = 1000$ obtained through Monte Carlo simulation, for $u = 1000$, 500 and 20. The figure also shows the curves of the approximation to $Z_{i,u}$ obtained using the model in this section. We can observe how the proposed method is able to track the distribution of $Z_{i,u}$ very accurately for $u = k$ and $u = 500$. However, at the end of the triangulation process a divergence appears as it can be observed for $u = 20$ in Figure 10 (c). The source of this divergence could, in large part, be attributed to the independence assumption made in (18). As the number of active input symbols u decreases, the dependence among the different $Z_{i,u}$ becomes stronger, and the independence assumption approximation falls apart.

Figure 11 shows the average number of inactivations needed to complete decoding for a linear random fountain code (LRFC)[7] and a RSD, both with average output degree $\bar{\Omega} = 12$ and $k = 1000$. The figure shows results obtained by Monte Carlo simulation and also the estimation of the number of inactivations obtained under our Poisson approximation. A tight match between simulation results and the estimation can be observed. The experimental results indicate that, although

[7]The degree distribution of a LRFC follows a binomial distribution with parameters k and $p = 1/2$ (see [42]).

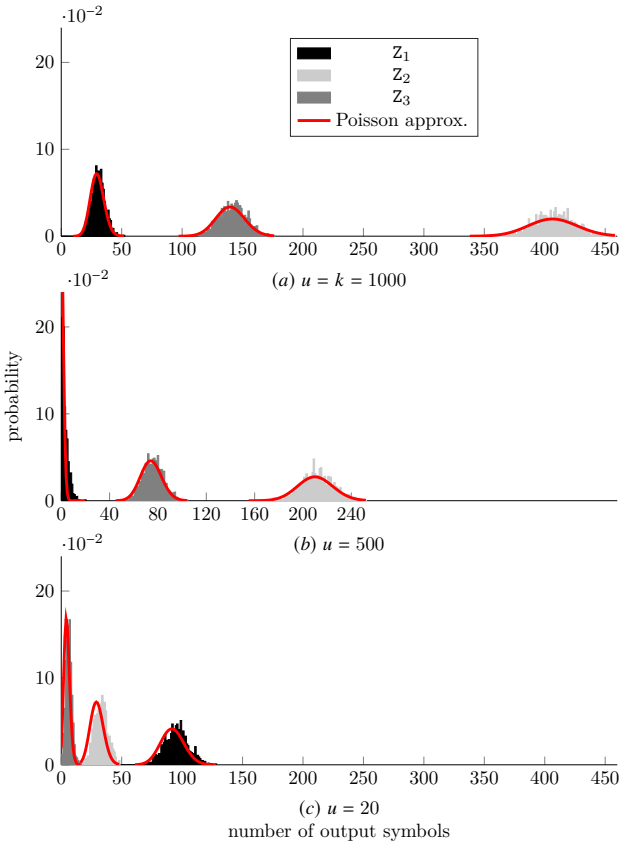


Fig. 10. Distribution of $Z_{1,u}$, $Z_{2,u}$ and $Z_{3,u}$ for an LT code with robust soliton distribution and $k = 1000$ obtained through Monte Carlo simulation. The upper, middle and lower figures represent respectively the distribution for $u = 1000$, 500 and 20. The black bars represent $Z_{1,u}$, the light grey bars $Z_{2,u}$ and the dark grey bars $Z_{3,u}$. The red lines represent the Poisson distribution approximation to $Z_{d,u}$ obtained employing the model in this section.

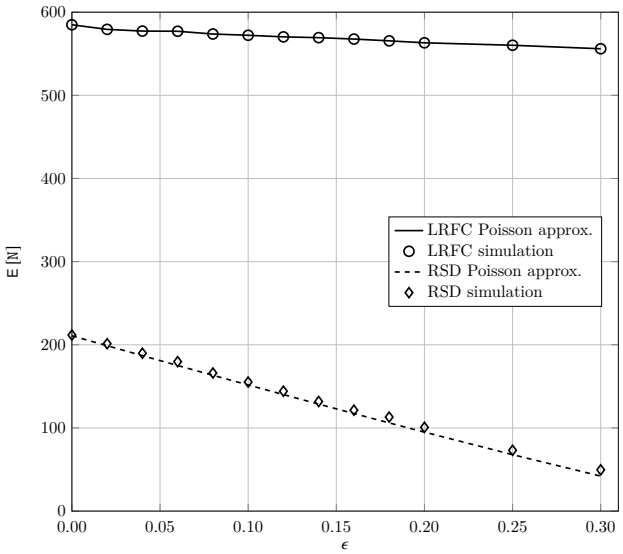


Fig. 11. Average number of inactivations needed to decode a linear random fountain code and a RSD for $k = 1000$ and average output degree $\bar{\Omega} = 12$. The markers represent simulation results and the lines represent the predicted number of inactivations using the proposed Poisson approximation.

the independence assumption made does not hold in general, it is a good approximation for most of the decoding process, deviating from simulation results only at the last stages of decoding. Thus, the proposed method can still provide a good approximation of the number of inactivations needed for decoding.

REFERENCES

[1] F. Lázaro, G. Liva, and G. Bauch, "Inactivation decoding analysis for LT codes," in *Proc. 52nd Annu. Allerton Conf. on Commun., Control, and Computing*, Monticello, Illinois, USA, Oct. 2015.

[2] F. Lázaro Blasco, G. Liva, and G. Bauch, "LT code design for inactivation decoding," in *Proc. 2014 IEEE Inf. Theory Workshop*, Hobart, Tasmania, Australia, Nov. 2014, pp. 441–445.

[3] J. Metzner, "An improved broadcast retransmission protocol," *IEEE Trans. Commun.*, vol. 32, no. 6, pp. 679–683, Jun 1984.

[4] J. Byers, M. Luby, and M. Mitzenmacher, "A digital fountain approach to reliable distribution of bulk data," *IEEE J. Select. Areas Commun.*, vol. 20, no. 8, pp. 1528–1540, Oct. 2002.

[5] M. Luby, "LT codes," in *Proc. 43rd Annual IEEE Symp. on Foundations of Computer Science*, Vancouver, Canada, Nov. 2002, pp. 271–282.

[6] P. Maymounkov, "Online codes," Technical report, New York University, Tech. Rep., 2002.

[7] A. Shokrollahi, "Raptor codes," in *Proc. of the 2004 IEEE Int. Symp. on Inf. Theory*, Chicago, Illinois, US, Jun. 2004, p. 36.

[8] M. Shokrollahi, "Raptor codes," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2551–2567, Jun. 2006.

[9] R. Karp, M. Luby, and A. Shokrollahi, "Finite length analysis of LT codes," in *Proc. 2004 IEEE International Symp. on Inf. Theory*, Chicago, Illinois, US, Jun. 2004.

[10] E. Maneva and A. Shokrollahi, "New model for rigorous analysis of LT-codes," in *Proc. 2006 IEEE International Symp. on Inf. Theory*, Seattle, Washington, US, Jul. 2006, pp. 2677–2679.

[11] S. Puducheri, J. Kliewer, and T. E. Fuja, "The design and performance of distributed LT codes," *IEEE Trans. Inf. Theory*, vol. 53, no. 10, pp. 3740–3754, Oct 2007.

[12] E. Hyytia, T. Tirronen, and J. Virtamo, "Optimal degree distribution for LT codes with small message length," in *IEEE Infocom*, Anchorage, USA, May 2007, pp. 2576–2580.

[13] D. Vukobratovic, C. Stefanovic, V. Crnojevic, F. Chiti, and R. Fantacci, "Rateless packet approach for data gathering in wireless sensor networks," *IEEE J. Select. Areas Commun.*, vol. 28, no. 7, pp. 1169–1179, Sep. 2010.

[14] G. Maatouk and A. Shokrollahi, "Analysis of the second moment of the LT decoder," *IEEE Trans. Inf. Theory*, vol. 58, no. 5, pp. 2558–2569, May 2012.

[15] M. Shirvanimoghaddam, Y. Li, S. Tian, and B. Vucetic, "Distributed raptor coding for erasure channels: Partially and fully coded cooperation," *IEEE Trans. Commun.*, vol. 61, no. 9, pp. 3576–3589, Sep. 2013.

[16] A. Shokrollahi, "Theory and applications of Raptor codes," *Mathknow*, vol. 3, pp. 59–89, 2009.

[17] "Technical Specification Group Services and System Aspects; Multimedia Broadcast/Multicast Service; Protocols and Codecs," Jun. 2012, 3GPP TS 26.346 V11.1.0.

[18] M. Shokrollahi, S. Lassen, and R. Karp, "Systems and processes for decoding chain reaction codes through inactivation," Feb. 2005, uS Patent 6,856,263.

[19] A. Shokrollahi and M. Luby, *Raptor Codes*. Foundations and Trends in Commun. and Inf. Theory, Now Publishers Inc., 2011.

[20] C. Lanczos, "Solution of systems of linear equations by minimized iterations," *J. Res. Nat. Bureau of Standards*, vol. 49, pp. 33–53, 1952.

[21] E. R. Berlekamp, *Algerbraic Coding Theory*. McGraw-Hill, 1968.

[22] B. A. LaMacchia and A. M. Odlyzko, "Solving large sparse linear systems over finite fields," *Advances in Cryptology-CRYPT090*, pp. 109–133, 1991.

[23] T. J. Richardson and R. L. Urbanke, "Efficient encoding of low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 638–656, Feb 2001.

[24] D. Burshtein and G. Miller, "An efficient maximum likelihood decoding of LDPC codes over the binary erasure channel," *IEEE Trans. Inf. Theory*, vol. 50, no. 11, nov 2004.

[25] E. Paolini, G. Liva, B. Matuz, and M. Chiani, "Maximum likelihood erasure decoding of LDPC codes: Pivoting algorithms and code design," *IEEE Trans. Commun.*, vol. 60, no. 11, pp. 3209–3220, Nov. 2012.
[26] B. Schotsch, H. Schepker, and P. Vary, "The performance of short random linear fountain codes under maximum likelihood decoding," in *Proc. 2011 IEEE International Conf. on Commun.*, Kyoto, Japan, Jun. 2011.
[27] B. Schotsch, R. Lupoaie, and P. Vary, "The Performance of Low-Density Random Linear Fountain Codes over Higher Order Galois Fields under Maximum Likelihood Decoding," in *Proc. 48th Annu. Allerton Conf. on Commun., Control, and Computing*, Monticello, Illinois, US, Oct. 2011.
[28] B. Schotsch, G. Garrammone, and P. Vary, "Analysis of LT codes over finite fields under optimal erasure decoding," *IEEE Commun. Lett.*, vol. 17, no. 9, pp. 1826–1829, Sep. 2013.
[29] B. E. Schotsch, "Rateless coding in the finite length regime," Ph.D. dissertation, Inst. of Commun. Systems and Data Proc., RWTH Aachen, Aachen, Germany, Jul. 2014.
[30] F. Lázaro, E. Paolini, G. Liva, and G. Bauch, "Distance spectrum of fixed-rate Raptor codes with linear random precoders," *IEEE J. Select. Areas Commun.*, vol. 34, no. 2, pp. 422–436, Feb 2016.
[31] C. Di, D. Proietti, I. Telatar, T. Richardson, and R. Urbanke, "Finite-length analysis of low-density parity-check codes on the binary erasure channel," *Information Theory, IEEE Transactions on*, vol. 48, no. 6, pp. 1570 –1579, jun 2002.
[32] G. Liva, E. Paolini, and M. Chiani, "Bounds on the error probability of block codes over the q-ary erasure channel," *IEEE Trans. Commun.*, vol. 61, no. 6, pp. 2156–2165, Jun. 2013.
[33] F. Lázaro, G. Liva, E. Paolini, and G. Bauch, "Bounds on the error probability of Raptor codes," in *Proc. IEEE Globecomm*, Washington DC, USA, Dec. 2016.
[34] C. Measson, A. Montanari, and R. Urbanke, "Maxwell construction: The hidden bridge between iterative and maximum a posteriori decoding," *IEEE Trans. Inf. Theory*, vol. 54, no. 12, pp. 5277–5307, Dec. 2008.
[35] A. Ashikhmin, G. Kramer, and S. ten Brink, "Extrinsic information transfer functions: Model and erasure channel properties," *IEEE Trans. Inf. Theory*, vol. 50, no. 11, pp. 2657–2673, Nov. 2004.
[36] K. Mahdaviani, M. Ardakani, and C. Tellambura, "On Raptor code design for inactivation decoding," *IEEE Trans. Commun.*, vol. 60, no. 9, pp. 2377–2381, Sep. 2012.
[37] T.-C. Ng and S. Yang, "Finite-length analysis of BATS codes," in *Proc. of 2013 IEEE International Symp. on Network Coding, (NetCod)*, Calgary, Alberta, Canada, Jun. 2013.
[38] E. Paolini, G. Liva, B. Matuz, and M. Chiani, "Maximum likelihood erasure decoding of LDPC codes: Pivoting algorithms and code design," *IEEE Trans. Commun.*, vol. 60, no. 11, pp. 3209–3220, Nov. 2012.
[39] M. Luby, A. Shokrollahi, M. Watson, and T. Stockhammer, "RFC 5053: Raptor forward error correction scheme: Scheme for object delivery," IETF, Tech. Rep., Oct. 2007.
[40] S. Kirkpatrick, D. Gelatt, and M. Vecchi, "Optimization by simmulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.
[41] F. Lázaro Blasco, G. Liva, and G. Bauch, "Enhancing the LT component of Raptor codes," in *Proc. of the 10th International ITG Conf. on Systems, Commun. and Coding, SCC 2015*, Hamburg, Germany, Feb. 2015.
[42] G. Liva, E. Paolini, and M. Chiani, "Performance versus overhead for fountain codes over $\mathbb{F}_q$," *IEEE Comm. Letters.*, vol. 14, no. 2, pp. 178–180, 2010.